Ngo, Long and Boyd, Colin and Nieto, Juan Gonzalez (2010) ***Automating computational proofs for public-key-based key exchange***. In: Provable Security : Proceedings of The 4th International Conference on Provable Security (ProvSec 2010) , 13-15 October 2010, Hotel Equatorial Melaka, Malacca, Malaysia.

Automating Computational Proofs for Public-key-based Key Exchange*

Long Ngo, Colin Boyd, and Juan González Nieto

Information Security Institute, Queensland University Of Technology,
GPO Box 2434, Brisbane QLD 4001, Australia
{lt.ngo, c.boyd, j.gonzaleznieto}@qut.edu.au

Abstract. We present an approach to automating computationally sound proofs of key exchange protocols based on public-key encryption. We show that satisfying the property called *occultness* in the Dolev–Yao model guarantees the security of a related key exchange protocol in a simple computational model. Security in this simpler model has been shown to imply security in a Bellare–Rogaway-like model. Furthermore, the occultness in the Dolev–Yao model can be searched automatically by a mechanisable procedure. Thus automated proofs for key exchange protocols in the computational model can be achieved. We illustrate the method using the well-known Lowe–Needham–Schroeder protocol.

1 Introduction

Proving security of cryptographic protocols and verifying those proofs is a hard, time-consuming and error-prone task when done by hand. Many flaws in security proofs were found after the proofs have been accepted and published [11, 10]. As a consequence, automated proofs have been considered a promising solution.

Research on automated proofs in Dolev–Yao models [16] has a long history thanks to the model's simplicity. Although this simplicity means sacrificing the faithfulness of the model, such automated tools have brought many significant successes in finding flaws. A well-known example is the Needham–Schroeder public-key protocol that was believed to be secure for many years until an attack was found by Lowe [19]. On the other hand, such tools cannot guarantee security in a computational sense, since Dolev–Yao models do not capture computational attacks defined in the usual cryptographic models.

Some automatic approaches for more realistic computational models have been proposed recently, shedding some light on this problem. However, the number of approaches and the types of protocol they can be applied on is still limited.

Contribution. This paper shows how we can automate the verification of security of key exchange protocols based on public key encryption. The security proofs achieved are computationally sound. The basic idea is as follows.

* Research partially funded by the Australian Research Council through Discovery Project DP0773348

– Kudla and Paterson proposed a modular way to prove security of key agreement protocols [18]. According to their approach, a security proof in the BR2000 model [2] is reduced to the hardness of a *gap* problem if we can show that the security of a simpler protocol in a simpler (computational) model is reduced to hardness of the related *computational* problem.
– Cortier et al. showed that we can automate secrecy proofs of public-key-based protocols [12]. Their idea results in a mechanisable search procedure for a property called *occultness*. However, their work is based on Dolev-Yao model.
– Our contribution provides a link between these two results. We show that security of a public-key-based protocol which has been checked to be occult in the simpler computational model is reduced to the hardness of a computational problem, which we construct using an IND-CCA encryption scheme. Therefore, to check the security proof in the BR2000 model, we can use the automatic tool by Cortier et al. [12] to check occultness.

Related work. Research on computationally sound automated proofs for cryptographic systems falls into two broad directions: direct proofs on computational models and indirect proofs via Dolev–Yao models.

Direct approaches. Direct approaches reason on protocol specifications often written in a programming language that has a computational semantics. Courant et al. [14] designed a Hoare-style logic to verify computational invariants, e.g. indistinguishability. However, this work can verify IND-CCA security only of an encryption scheme that is constructed from a trapdoor permutation. Later, Gagné et al. [17] extended this work to symmetric block ciphers. Blanchet [5] designed a variant of π-calculus to formalise games, and developed CryptoVerif, a tool that can automatically transform games using game-hopping techniques, thereby freeing the human from the mundane parts of the proof. CryptoVerif can be potentially extended to cover many types of protocols, but it is hard to make it fully automated, i.e. manual guidance is required in non-trivial situations. Datta et al. [15] tuned the Computational Protocol Composition Logic for verifying key exchange protocols, resulting in security proofs in the Bellare-Rogaway model [3]. However, their work is limited to Diffie–Hellman-based protocols and although they claimed the work is mechanisable, they have not shown how to do it in details or provided an actual tool.

Indirect approaches. In contrast, indirect approaches exploit automated tools designed for verifying properties in Dolev–Yao models by showing in which cases symbolic properties imply computational ones. Cortier and Warinschi [13] proved the computational soundness of a Dolev–Yao model, by showing how to map between symbolic and computational traces. They also tested their idea by using Casrul [9], a Dolev–Yao-based tool to make a Dolev–Yao security proof. That work differs from ours in the models used. The Dolev-Yao model they used is the model for Carsul, a protocol verifier for a fixed number of sessions, while the Dolev–Yao model we use is the model for

Securify [12], a tool that gives proofs for an unbounded number of sessions and adversarial operations. In the other side, the computational model they designed is a more general and simpler, e.g. no session corruption, but the one we use, the BR2000 model [2], which is designed specifically for key exchanges.

Canetti and Herzog [6] used ProVerif [4] to automatically verify some symbolic criteria and showed that a protocol satisfying such criteria realises an ideal functionality for key exchange protocols. This work focuses on public-key-based key exchange protocols like our work, but the two results are not strictly comparable. Canetti and Herzog use the universal composability (UC) model with a UC-secure public-key encryption scheme which is a stronger security than we use since it allows security under composability. At the same time their model is weaker than ours because they do not model adaptive corruptions and session key reveals. We also note that Canetti and Krawczyk [8] have shown that universal composability can be obtained for 'free' for a slightly weaker functionality. Canetti and Herzog [6] proved that *strong secrecy* (an equivalence property), or *observational equivalence* between processes that have different values for a secret variable, implies UC-security. We prove that occultness, which means *standard secrecy* (a trace property and a less strong notion), implies security in the Bellare–Rogaway model.

Later, Canetti and Gajek extended the work [7] to deal with key exchange based on key encapsulation and signature schemes. With this type of protocols, they consider adaptive corruptions, session key reveals, and forward secrecy in the model. The authors also showed that the plain Diffie-Hellman protocol realizes their key encapsulation functionality, thus the result is widely applicable on a number of Diffie-Hellman based key exchange protocols.

2 Preliminaries

2.1 Gap Problem

Gap problems were first mentioned by Okamoto and Pointcheval [22]. We summarize the idea here. Let $f : X \times Y \to \{0, 1\}$ be any relation on sets X and Y.

- The *computational* problem of f is: given $x \in X$, find any $y \in Y$ such that $f(x, y) = 1$ if such a y exists, otherwise return Fail.
- The *decisional* problem of f is: given $(x, y) \in X \times Y$, to decide if $f(x, y) = 1$ or not.

Definition 1. *The gap problem of f is: given the decisional oracle of f, to solve the computational problem.*

2.2 A Modular Proof for Key Agreement

Kudla and Paterson proposed a modular way to prove security of key exchange protocols in a modified Bellare-Rogaway (mBR) model [18]. We use their idea in this work with a little simplification, in that we do not consider any corrupted oracle to be a *fresh* one because we do not model key compromise impersonation attacks. We call our mode mBR′ model to differentiate it from Kudla and Paterson's.

The mBR′ Game. Denote the set of participants IDs as $\mathcal{U}$ and assume each participant $U \in \mathcal{U}$ has a public key P_U and a private key S_U. We use Π_U^i to denote the oracle of the *ith* instance of U. An oracle Π_U^i may accept once at any time. After that it holds a role $role \in \{initiator, responder\}$, a partner ID pid, a session ID sid and a session key $seskey$. Each oracle follows the protocol rules and responds to input messages from the adversary. Each oracle Π_U^i also stores a public transcript $T_{\Pi_U^i}$ that records all messages sent and received by that oracle.

The game is played between a challenger C and an adversary E. C runs a Setup algorithm on security parameter k, generating public parameters, a set of participants $\mathcal{U}$ and oracles $\{\Pi_U^i\}$, distributing long-term keys to participants, and selecting a bit b. E is also given all public keys and access to all oracles, including random oracles.

Adversarial Queries. The adversary can make the following queries.

- **Send**(U, i, M): E gives the oracle Π_U^i a message M. If this oracle's $pid = U'$, then Π_U^i assumes that M is from U' and acts according to the protocol. For initiating oracles, E can make a special **Send** query λ, which tells Π_U^i to set $role_U = initiator$. If Π_U^i did not receive a message λ as the first message, $role_U$ will be $responder$.
- **Reveal**(U, i): E uses this query to obtain the session key of Π_U^i (if any).
- **Corrupt**(U): This allows E to learn U's long-term key.

Oracle States. An oracle Π_U^i can be in the following states.

- **Accepted**: An oracle is in this state if it has received a properly constructed messages to make a session key and the oracle accepts the key.
- **Rejected**: An oracle is in this state if it decides not to establish a session key and abort the protocol.
- **Revealed**: An oracle is in this state if it has answered a Reveal query.
- **Corrupted**: An oracle is in this state if U has answered a Corrupt query.

Partnership. Two oracles Π_U^i, holding $(seskey, sid, pid)$ and $\Pi_{U'}^j$, holding $(seskey', sid', pid')$ are said to be *partners* if they have accepted and:

1. $sid = sid'$, $seskey = seskey'$, $pid = U'$, $pid' = U$;
2. $role_U = initiator$ and $role_{U'} = responder$ or vice versa;
3. no other oracle has accepted with session ID equals sid.

Freshness. An oracle Π_U^i is *fresh* if it and its partner $\Pi_{U'}^j$ (if any) are not revealed and neither U nor U' is corrupted.

Test Query. After E has made a polynomial number of queries in k, E can make a **Test** query to an oracle Π_U^i, which must be accepted and fresh. If $b = 0$ then Π_U^i outputs a randomly chosen session key $seskey_{random}$, otherwise it outputs its real session key $seskey_{\Pi_U^i}$.

After that, E can continue querying, but not reveal or corrupt the test oracle or its partner. Finally, E outputs his guess b' for b. E's advantage, denoted $Advantage^E(k)$, is $|1/2 - \mathsf{Pr}[b' = b]|$.

Definition of Security. A *benign adversary* is one who just relays messages between parties without any modification. Then the security definition for authenticated key exchange (AKE) is defined as follows.

Definition 2. *A protocol is an mBR′-secure AKE protocol if:*

1. *in the presence of a benign adversary, two oracles running the protocol accept and hold the same session key and session ID, and the session key is distributed uniformly at random on* $\{0,1\}^k$*; and*
2. *for any adversary* E*,* $Advantage^E(k)$ *is negligible.*

The cNR-mBR′ Model. The cNR-mBR′ model is the same as mBR′, except:

- the adversary cannot make any **Reveal** query;
- instead of a normal **Test** query, the adversary selects an accepted and fresh oracle Π_U^i and outputs a guess $seskey$ for the oracle's session key $seskey_{\Pi_U^i}$. Then $Advantage^E(k) = \mathsf{Pr}[seskey = seskey_{\Pi_U^i}]$.

Following the technique of Kudla and Paterson [18], a protocol Π defined in the mBR′ model can be first proven secure in the cNR-mBR′ model, which is simpler. We will define a compiler to promote such a protocol to one secure in the mBR′ model as long as the protocol Π produces a *session string* ss_Π and uses a hash function, which is modelled as a random oracle, to finally compute a *hashed session key.* We also use Kudla and Paterson's notion of strong partnering.

If a protocol has strong partnering, the adversary cannot trivially win the game by making two oracles, which are not partners, have the same session key and then using the **Reveal** query. Strong partnering can always be achieved by including partnering information in the session string; specifically we add the session identifier and the identity of the initiator and responder to the session string.

Definition 3 ([18]). *If* Π *is a key exchange protocol and there exists an adversary* E*, who plays the mBR′ game with* Π*, and with non-negligible probability (in security parameter* k*) can make any two oracles* Π_U^i *and* $\Pi_{U'}^j$ *accept and hold the same session key when they are not partners, then we say that* Π *has* weak partnering*. Otherwise* Π *has* strong partnering*.*

Definition 4. *Suppose Π is a key exchange protocol. The* session string decisional problem *for protocol Π is: given an oracle Π_U^i and its transcript T_U^i in the mBR$'$ model, public keys P_U and $P_{U'}$ (where $pid_U^i = U'$) and s, to decide whether s is the session string of Π_U^i or not.*

In order to use the cNR-mBR$'$ model, given a protocol Π, we define a protocol π to be the same as Π, except that the session key of π is the session string of Π.

Theorem 1. *Suppose that a key exchange protocol Π uses a hash function H to compute a hashed session key on completion of the protocol and Π has strong partnering. If the cNR-mBR$'$ security of the related protocol π is probabilistic polynomial time reducible to the hardness of the computational problem of some relation f, and the session string decisional problem for Π is polynomial time reducible to the decisional problem of f, then the mBR$'$ security of Π is probabilistic polynomial time reducible to the hardness of the gap problem of f, assuming that H is a random oracle.*

Proof. The proof for the original mBR model is given by Kudla and Paterson [18]. The proof for mBR$'$ is essentially the same. The idea of the proof is simple. Assume that there is an adversary A, given an algorithm E that can win cNR-mBR game, can solve the computational problem. Now we have to show that we can construct an algorithm B, given the decisional problem oracle and an adversary D that can win mBR game, can solve the computational problem. Thus, the heart of the idea is to use D and the decisional problem oracle to simulate E, and make A solve the computational problem for B.

Notice that in our model mBR$'$ (hence in cNR-mBR$'$ also) we do not consider a corrupted oracle to be a fresh one. However, while simulating E, B passes all **Corrupt** queries from D to A blindly, therefore this difference does not matter.

2.3 Chosen Ciphertext Security in the Multi-user Setting

For the simulation in our proof (see Section 3.3), we need an encryption scheme that is secure even if the adversary can ask for more than one challenging ciphertext encrypted by more than one public key. This is call indistinguishability under chosen ciphertext attack in the multi-user setting (IND-CCA-M) [1]. Fortunately, adaptive CCA security also implies such kind of security.

Definition 5 (IND-CCA-M).
For all equal-length strings m_0, m_1 and any $b \in \{0,1\}$, the left or right selector LR *is defined as*

$$\mathrm{LR}(m_0, m_1, b) = m_b.$$

For a bit b that is unknown to the adversary, a LR *encryption oracle $\mathcal{E}_{pk}\mathrm{LR}(\cdot,\cdot,b)$, given query (m_0, m_1) where m_0, m_1 are two equal-length plaintexts, first sets $m_b \leftarrow \mathrm{LR}(m_0, m_1, b)$, then outputs the encryption of m_b using the public key pk.*

A decryption oracle $\mathcal{D}_{sk}(\cdot)$, given a valid ciphertext c, outputs the corresponding decryption of c using the secret key sk.

We have the following experiment. Let $\mathcal{PE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a public-key encryption scheme. The adversary, A_{cca}, has access to n LR *encryption oracles $\mathcal{E}_{pk_i}\mathrm{LR}(\cdot,\cdot,b)$ and n corresponding decryption oracles $\mathcal{D}_{sk_i}(\cdot)$, where A_{cca} is not allowed to query $\mathcal{D}_{sk_i}(\cdot)$ on an output of $\mathcal{E}_{pk_i}\mathrm{LR}(\cdot,\cdot,b)$. Let I be some initial information string. For $b \in \{0,1\}$, we have*

$\mathbf{Exp}^{n-cca}_{\mathcal{PE},I}(A_{cca}, b)$
  For *i= 1, . . . , n* ***do*** $(pk_i, sk_i) \leftarrow \mathcal{K}(I)$ ***EndFor***
  $d \leftarrow A_{cca}^{\mathcal{E}_{pk_1}\mathrm{LR}(\cdot,\cdot,b),\ldots,\mathcal{E}_{pk_n}\mathrm{LR}(\cdot,\cdot,b),\mathcal{D}_{sk_1}(\cdot),\ldots,\mathcal{D}_{sk_n}(\cdot)}(I, pk_1, \ldots, pk_n)$
  Return d

The advantage of A_{cca} is defined as

$$\mathsf{Adv}^{n-cca}_{\mathcal{PE},I}(A_{cca}) = \Pr[\mathbf{Exp}^{n-cca}_{\mathcal{PE},I}(A_{cca}, 0) = 0] - \Pr[\mathbf{Exp}^{n-cca}_{\mathcal{PE},I}(A_{cca}, 1) = 0]$$

We say that $\mathcal{PE}$ is secure against chosen ciphertext attack in the multi-user setting *if for all A_{cca} and polynomial n, $\{\mathsf{Adv}^{n-cca}_{\mathcal{PE},I}(A_{cca})\}$ is negligible.*

Lemma 1 ([1]). *If an encryption scheme is IND-CCA secure then it is IND-CCA-M secure.*

2.4 An Automatic Search Procedure in Dolev Yao model

Cortier et al. [12] proposed an automatic procedure that checks whether or not a protocol has the property called *occult*, which has been proved to imply secrecy in the Dolev–Yao model. We will briefly introduce the idea here.

Message Fields. Fields is the set of messages, which can be either primitive or compound fields. A primitive field's type can be one of *Agent*, *Key*, *Nonce*. *Key* and *Nonce* make the set *Basic*, the only set in which a field can be designated as secret. As a notational convention, variables A, B and variants denote agents; K and variants denotes keys.

Each agent A has a public key $\mathsf{pub}(A)$ and the related private-key $\mathsf{prv}(A)$. Each key K has an inverse key K^{-1}, i.e. $\mathsf{pub}(A)^{-1} = \mathsf{prv}(A)$ and $\mathsf{prv}(A)^{-1} = \mathsf{pub}(A)$.

Events and Global States. The system state is represented by a set of ordered events. There are three kinds of events: *messages*, *spells* and *states*.

- A message event is just a field that is the content of a sent message.
- A spell event $C = (S, L) \in Spells$ where the book $Book(C) = S$, is a set of basic secrets shared among a set of agents $Cabal(C) = L$.
- A state event is of the form $\mathsf{A}_n(X)$ where A is a role, n is the protocol step of role A and X is the concatenated field in memory held by the state. The set of *basic secrets* of a spell is made up by its book and long-term (private) keys of its cabal.

$$Sec(C) = Book(C) \cup ltk(Cabal(C))$$

A *global state* is a set of events. The *content* of a global state H is its set of messages

$$Cont(H) \stackrel{\text{def}}{=} H \cap Fields$$

A basic field is *unused* in H if it is neither a part of any field in $Cont(H)$ nor $Sec(H)$.

$$unused(H) \stackrel{\text{def}}{=} \{X \in Basic | X \notin \mathsf{parts}(Cont(H)), X \notin (Sec(H))\}$$

where $\mathsf{parts}(.)$ is defined in the following paragraph.

Inductive Relations.

- $\mathsf{parts}(S)$ is the set of all sub-fields of fields in S (not including keys of encryptions).
- $\mathsf{analz}(S)$ is the subset of $\mathsf{parts}(S)$ having only subfields that are accessible to adversary.
- $\mathsf{synth}(S)$ is the set of all fields constructible from S by concatenation and encryption using fields and keys in S.
- $\mathsf{fake}(S) \stackrel{\text{def}}{=} \mathsf{synth}(\mathsf{analz}(S))$.

Ideals and Coideals.

- An *ideal* $\mathcal{I}$ is the set of fields that must be protected in order to protect secrets in S. It is the smallest superset of S such that the concatenation $[X, Y] \in \mathcal{I}(S)$ if $X \in \mathcal{I}(S)$ and $Y \in \mathcal{I}(S)$, and $\{X\}_K \in \mathcal{I}(S)$ if $X \in \mathcal{I}(S)$ and $K^{-1} \notin \mathcal{I}(S)$.
- The coideal $\mathcal{C}(S)$ is the complement of $\mathcal{I}(S)$.

Protocols. A protocol specification is made up by a set of transitions. A transition is of the form $Pre(t) \stackrel{New(t)}{\longrightarrow} Post(t)$, where $Pre(t)$ and $Post(t)$ are sets of events and $New(t)$ is a new set of nonces.

Except for the initialisation transition, a transition t shows a state change of one role. A message or post spell (but not both) may be introduced in $Post(t)$.

One restriction is that secrets in a post spell must be in $New(t)$. One condition for protocol security is *regularity*, implying that there is no long-term key introduced into a post message.

Global State Transitions. Given a protocol P and a set of initial knowledge I of the adversary, the *global succession* relation defines how a state H is transformed to a new state H' as follows.

- H' is an *honest* successor of H, if there is an applicable transition t in P such that $H' = (H \backslash (Pre(t) \cap States)) \cup Post(t)$. A transition t is *applicable* in H if $Pre(t) \subseteq H$ and $New(t) \subseteq unused(H)$
- H' is a *fake* successor of H, if there exists a field $X \in \mathsf{fake}(Cont(H) \cup I)$ such that $H' = H \cup \{X\}$.

The set of reachable states from P and I is denoted by $reachable(P, I)$.

Requirement for Secrecy. A spell is *compatible* with initial knowledge I if I does not have any of its basic secrets.

$$compatible(I) \stackrel{\text{def}}{=} \{C | Sec(C) \cap \mathsf{parts}(I) = \emptyset\}$$

Given adversarial initial knowledge I, a global state H is called *I-discreet* if $Cont(H) \subseteq \mathcal{C}(Sec(C))$ for any *I-compatible* spell $C \in H$.

Definition 6. *A P-configuration is a tuple (I, H, C) in which $H \in reachable(P, I)$, H is I-discreet, $C \in compatible(I)$, and $C \in H$. A protocol P is occult if for all P-configuration(I, H, C) and for every transition $t \in P$,*

$$Cont(Post(t)) \subseteq \mathcal{C}(Sec(C)).$$

Millen et al. [21] give a secrecy theorem saying that if a protocol is *occult* then it provides secrecy (in a Dolev–Yao style model). More importantly, Cortier et al. [12] proposed an automatic search procedure to check if a protocol has *occultness*. This result is very important for our work, because later we will show that if a protocol has occultness and passes some simple checks, that protocol is also secure in the cNR-mBR′ model.

3 Security in the cNR-mBR′ Model

In this section, we show that the property *occultness* also implies security in the cNR-mBR model under a condition that both initiator and responder provide nonces for building the session key. We consider only protocols that are based on public-key encryption, i.e. every message includes only fields that can be nonces, party IDs, concatenation or encryption of other fields.

Definition 7. *A two-party protocol is said to be based on public-key encryption if every message m is constructed by the following syntax:*

$$m ::= \mathsf{nonce} \mid \mathsf{partyID} \mid \mathsf{Enc}_{pk}(m) \mid \mathsf{concat}(m, m),$$

where nonce *is a random number,* partyID *is a party ID number,* $\mathsf{Enc}_k(.)$ *is encryption under the public key pk and* $\mathsf{concat}(.,.)$ *is concatenation of two fields.*

3.1 Occultness Property in cNR-mBR′ Model

In this section we link Dolev-Yao occultness property with some property in our computational models. Those computational properties will be useful for establishing security proofs in the cNR-mBR′ model later.

Lemma 2. *If a protocol π has the occult property, then any secret nonce sent between parties is always encrypted.*

Proof. Informally, *occultness* implies secrecy (in Dolev-Yao models), therefore it must also imply that no secret nonce is sent in the plain form (otherwise the adversary can learn it easily).

Assume that there is transition t, where $Post(t)$ contains a message event whose fields include a nonce r (in the plain form). Because r must be kept secret, there must be a spell event C where $r \in Sec(C)$. This means $r \in \mathcal{I}(Sec(C))$, i.e. $r \notin \mathcal{C}(Sec(C))$. Therefore, $Post(t) \notin \mathcal{C}(Sec(C))$, i.e. $Cont(Post(t)) \notin \mathcal{C}(Sec(C))$. But this means the protocol is not occult.

Lemma 3. *Suppose a protocol π has the occult property in the Dolev-Yao model, and the underlying public key encryption scheme is IND-CCA secure. Consider any oracle Π^i_Z in the cNR-mBR$'$ model, whose pid is V, where Z and V have not been corrupted. Then with a negligible probability a ciphertext of a nonce created by Π^i_Z appears (in a transcript), where that ciphertext is not made either by an oracle of Z whose pid is V or by an oracle of V whose pid is Z.*

Proof (Sketch). Because of the scope of this paper, we will explain the outline of the proof. This lemma is only for a specific case of linking two trace properties, which are enough for our work, between Dolev-Yao and computational models. The idea of proof is based on the trace mapping technique, which has been fully demonstrated by Micciancio and Warinschi [20] for the general case (the models they used are very similar our models here). The intuitive idea behind the proof is: if the adversary is the first to make such a ciphertext, then we can break IND-CCA security; on the other hand, if any party oracle is the first to make such a ciphertext, then the protocol is not occult.

Assume the Lemma is false, then let c be the first such ciphertext. There are two cases: c is created by the adversary or c is created by a party oracle.

First we examine the former case, i.e. there is an adversary A who plays the cNR-mBR$'$ can make c with a non-negligible probability. Now we show that we can construct an algorithm G that can have non-negligible advantage in the experiment $\mathbf{Exp}^{2-cca}_{\mathcal{PE},I}(A_{cca}, b)$ (see Section 2.3) as follows.

- G creates a cNR-mBR$'$ game, and choose two parties Z and V to replace their public keys with the public keys from the experiment $\mathbf{Exp}^{2-cca}_{\mathcal{PE},I}(A_{cca}, b)$.
- Because G has public keys and decryption oracles of Z and V and all of public and secret keys of other parties, the cNR-mBR$'$ game can be simulated perfectly except for either an oracle of Z, whose *pid* is V or an oracle of V, whose *pid* is Z.
- For such an oracle, G prepares two sets of nonces. While simulating it, G always uses the corresponding encryption oracle $\mathcal{E}_{pk_i}\mathrm{LR}(\cdot, \cdot, b)$ whenever G needs encrypting. Notice that G never has to encrypt any of those nonces under a public key not of Z and V, because we are assuming the adversary is the first one to do it. In addition, G never has to use a plain nonce, but its encryption, according to Lemma 2. Therefore, even G does not know which set of nonce is actually used, but the simulation is still perfect, until c appears.
- A can still corrupt any party but not Z and V.

– When c appears, G submits it to the decryption oracle of Z, then G can guess the bit b correctly.

Second, lets examine the latter case, i.e. a party oracle Π_Y^j, whose *pid* is X, is the first to break the property by creating c under the public key of X, where $\{Y, X\}$ is not $\{Z, V\}$ or $\{V, Z\}$. Notice that Y must be either Z or V, because before Π_Y^j makes c, Π_Y^j must have received another encryption from Z or V [1]. We show that it is possible to construct a situation in the Dolev-Yao model, in which the protocol is not occult. First, for every bitstring value, including nonces and party IDs, we map it to a symbolic value. Now, in the Dolev-Yao model, we start with a situation when all parties are corrupted but Z, V. The Dolev-Yao adversary has to make the same transcripts as he did in the computational model, but now with symbolic values. We have to make sure that he is able to do that. Obviously, there is no problem for creating transcripts between Z or V with another party, because that party is corrupted. For any thing in any transcript between Z and V, if that is related to any nonce created by Z and V, then the Dolev-Yao adversary must be able to make the symbolic version, because he has corrupted all other parties except Z and V. In the other case, if that thing is related to a nonce created by an oracle of Z, whose *pid* is V and vice versa, we argue that the Dolev-Yao adversary is also able to create the symbolic version. Assume the contrary, then there is a bitstring that the Dolev-Yao adversary can not make the symbolic version. Then, in the experiment $\mathbf{Exp}^{2-cca}_{\mathcal{PE},I}(A_{cca}, b)$ above, given that bistring we can always use the decryption oracles to recursively decrypt ciphertexts. For any ciphertext that is not allowed to be decrypted by the decryption oracles, we skip it. There must be a ciphertext, that can be decrypted by the decryption oracles, giving us the hidden nonce in $\mathbf{Exp}^{2-cca}_{\mathcal{PE},I}(A_{cca}, b)$ (if there is no such a ciphertext, there would not be any problem for the Dolev-Yao adversary to make symbolic versions of transcript).

Now we have a situation in the Dolev-Yao model, where we have an instance of party Y sends out a ciphertext of a nonce, which has been created by Z, under the public key of X (where X is not Z). But this means the protocol is not occult.

3.2 Hard Problems Used for Security Proofs

According to the modular approach [18] that we are following, we need a set of computational, decisional and gap problems, which must be hard. Now we define the following problems and show that IND-CCA implies the hardness of them.

Informally, the computational problem we are going to define is based on the encryption property one-wayness under adaptively chosen ciphertext attack in the multi-user setting (OW-CCA-M). In this attack, we ask the adversary to find the plaintext of a random ciphertext, while allowing him to get the ciphertext of any message related to the hidden message, in a two-user setting. And, because

[1] otherwise Π_Y^j has no information about nonces in c, because the adversary has not faked any such an encryption

IND-CCA implies IND-CCA-M, we just have to show that IND-CCA-M implies OW-CCA-M. Then the decisional and gap problems are defined accordingly.

Definition 8. *Given a public-key encryption scheme $\mathcal{PE}$ with plaintext and ciphertext spaces $M_{\mathcal{PE}}$ and $C_{\mathcal{PE}}$, and a pair of public and secret keys (pk_z, sk_z), we define the following relation f:*

$$f : (M_{\mathcal{PE}} \times C_{\mathcal{PE}}) \rightarrow \{0,1\},$$

$$\text{where } f(m,c) = \begin{cases} 1 \text{ if } \mathcal{D}_{sk_z}(c) = m \\ 0 \text{ otherwise} \end{cases}$$

Now we can define our problems.

Let $\mathcal{E}_{pk_z}\mathrm{I}(\cdot,\cdot,\cdot)$ be an "inserting" encryption oracle of pk_z, which on input $(m_1, \mathcal{E}_{pk_z}(m_2), m_3)$, outputs $\mathcal{E}_{pk_z}(m_1, m_2, m_3)$; and $\mathcal{E}^{pk_v}_{pk_z}\mathrm{C}(\cdot)$ be a "converting" encryption oracle from pk_z to pk_v, which on input $\mathcal{E}_{pk_v}(m)$, outputs $\mathcal{E}_{pk_z}(m)$.

The adversary is given the public key pk_z, an additional public key pk_v and all corresponding decryption, inserting and converting oracles of pk_z and pk_v, where the decryption oracles never answer if input is from the inserting or converting oracles. We have the following problems of $\mathcal{PE}$.

- ***Computational problem**: The adversary is given c to compute m such that $f(m,c) = 1$.*
- ***Decisional problem**: The adversary is given c and m to determine if $f(m,c) = 1$ or not.*
- ***Gap problem**: Given an oracle that can solve the decisional problem above and c, to compute m such that $f(m,c) = 1$.*

Here the decryption oracle $\mathcal{D}_{sk_z}(\cdot)$ never decrypts c or any ciphertext from any oracles.

Lemma 4. *If the underlying encryption scheme is IND-CCA, the problems in Definition 8 are hard.*

Proof. We have shown that IND-CCA implies IND-CCA-M (see Section 2.3). Therefore, what we have to show is that solving any of the problems above is at least as hard as winning the IND-CCA-M game.

Given an algorithm E that can solve one of the problems above, we construct an algorithm F that can have non-negligible advantage in the experiment $\mathbf{Exp}^{2-cca}_{\mathcal{PE},I}(A_{cca}, b)$ (see Section 2.3).

The construction is as follows. F picks a pair of messages (m_0, m_1) randomly, then submits them to $\mathcal{E}_{pk_1}\mathrm{LR}(\cdot,\cdot,b)$ and then forwards the output c as the challenge ciphertext to E. Now, F has to simulate all necessary oracles that E needs. For any decryption request, F just forwards it to the decryption oracle in the experiment. For requests to inserting and converting oracles, because F can submit any message to oracles $\mathcal{E}_{pk_z}\mathrm{LR}$ and $\mathcal{E}_{pk_v}\mathrm{LR}$, F can also simulate those oracles.

We examine each case as follows.

– If E can solve the computational problem, i.e. output m_b, F can see m_b to guess b correctly.
– If E can solve the decisional problem, F just asks E if m_0 or m_1 is actually encrypted. Therefore F can guess b correctly.
– If E can solve the gap problem, F must simulate the decisional oracle. Given a request to the decisional oracle including a ciphertext c and a plaintext m, if c is not an output of any oracle $\mathcal{E}_{pk_z}\mathrm{LR}$, F can use $\mathcal{D}_{sk_z}(\cdot)$ to check the plaintext. Otherwise, if m is one of the two possible plaintexts of c (F must know them), F just outputs Yes. In this latter case, the probability that F simulates the decisional oracle wrongly is negligible, i.e. m is m_{1-b}, because c contains no information of m_{1-b}.
  Finally, F can see the output of E to guess b correctly.

3.3 Our Main Theorem

Theorem 2. *Suppose a protocol π is occult and based on an IND-CCA public-key encryption scheme, each side sends out at least one nonce, and the session key includes all exchanged nonces. Then the security of π in the cNR-mBR$'$ model is probabilistic polynomial time reducible to the hardness of the corresponding computational problem (according to Definition 8).*

Proof. Assume that there is an adversary A that participates in π in the cNR-mBR$'$ model and outputs the session key with a non-negligible probability ϵ_A in a time τ_A, where k is the security parameter. We will show that we can build an adversary B who solves the computational problem, i.e. given a ciphertext c, output the plaintext m of c using some oracles, with some probability $g(\epsilon_A)$ and in time $h(\tau_A)$ where g and h are polynomial functions.

The idea behind the reduction is as follows. B will try to make a session key of an oracle of Z, whose *pid* is V, to contain m. Without knowing m and the secret key used to decrypt c, B can still do that by using inserting and converting oracles to simulate the transcript of that oracle, as long as the oracle and its partner have not been corrupted. Fortunately, according to Lemma 3, B never has to simulate a ciphertext of m under another key except P_Z and P_V (otherwise the simulation fails because B does not know m). Finally, if A chooses that oracle to test, then the guess of the session key from A will help B to output m.

Now we formally describe how B works. As we have defined the computational problem of f, B is given the public key pk_z, which has been used to make c, an additional public key pk_v and all corresponding decryption, inserting and converting oracles. B makes the cNR-mBR$'$ game as follows.

– **Setting up:** B runs a $\mathsf{Setup}(k)$ algorithm to set up a set of participants $\{U\}$, their oracles and long-term keys for each participant as defined in Section 2.2. Then B chooses randomly two parties Z and V, replaces their public keys with pk_z and pk_v respectively. After that, B picks randomly a party oracle Π^j_Z, whose *pid* is V.
  Finally B gives all public keys to A.

– **Answering queries:**
  • Send(U, i, M):
    ∗ If this is the query for Π_Z^j, i.e. $U = Z$ and $i = j$, where after that Π_Z^j has to reply the first message containing a nonce (in an encrypted form, see Lemma 8), then Π_Z^j considers the hidden m as his first nonce and starts using corresponding inserting and converting oracles to make valid replies.
    Notice that if Π_Z^j has not been required to use any nonce, then the inserting and converting oracles have not been used any time.
    ∗ If no ciphertext inside M is from any inserting or converting oracles that B has been given (to solve the computational problem), then B can parse all the content M by using relevant secret keys or decryption oracles.
    Now there are two cases:
      · When B is parsing M, if there is a nonce that is supposed to be the hidden m, then B puts the party oracle Π_U^i in the state **Rejected**, because according to Lemma 3, M is an invalid message with an overwhelming probability.
      · Otherwise, B keeps simulating according to the protocol specification.
    ∗ If there is at least one ciphertext inside M that is from any inserting or converting oracles that B has been given, then certainly B knows what is the next state of Π_U^i (B knows m is inside, B just does not know what m is). But it maybe a problem if Π_U^i has to reply something containing m.
    Now, according to Lemma 3, B never have to make a ciphertext of m under a key different from P_Z or P_V. And to make a ciphertext of m under P_Z or P_V, B uses relevant inserting and converting oracles.
  • Corrupt(U):
    ∗ If $U = Z$ or $U = V$, then B stops using A and output randomly one bit b'.
    ∗ Otherwise B just gives A the corresponding private key.
– Finally, A must choose an accepted and fresh oracle and output his guess for the session key.
  Suppose the number of participants is n_{par} and each party may have n_{ses} sessions, where n_{par} and n_{ses} are polynomial functions of k.
  With a non-negligible probability $\frac{1}{n_{par}.n_{ses}}$, Π_Z^j is the oracle chosen by A. In this case, B just extracts all nonces from the session key outputted by A and outputs the nonce which is supposed to be m (Since the session key is made of nonces from both sides, it must contains m). Therefore, if Π_Z^j is chosen, the probability that B wins is $\eta_1 = \epsilon_A$.
  Otherwise, B just stops using A and outputs a random guess. Therefore, if Π_Z^j is not chosen, the probability that B wins is η_2 and negligible.
  Therefore, the probability that B wins is $\eta = \frac{1}{n_{par}.n_{ses}}.\eta_1 + (1 - \frac{1}{n_{par}.n_{ses}}).\eta_2 \geq \epsilon_A.\frac{1}{n_{par}.n_{ses}}$, in a time $h(\tau_A)$ where $h(.)$ is a polynomial function.

4 Automated Proofs in the mBR′ Model

4.1 The Session String Decisional Problem

In order to establish security in the mBR′ model from security in the cNR-mBR′ model, we need to show that while we are making a reduction from the gap problem to mBR′-security, we must be able to solve the session string decisional problem [18]. The following lemma shows that we can do it with any public-key-based protocol.

Lemma 5. *Given all the public and private keys used in an mBR′ game, except the private key of an arbitrary party Z, and the oracle for the decisional problem of f based on public and private key of Z, there exists a polynomial time algorithm S to solve the session string decisional problem in the mBR′ game.*

Proof. Since S has all public and private keys except the secret key of Z, S can open all ciphertexts with the exception of any ciphertext made under the public key of Z. However, for such a ciphertext, S can always use the decisional problem of f to check if a nonce is the plaintext or not. Therefore, S can always solve the session string decisional problem in the mBR′ game in polynomial time.

4.2 How to Automate Proofs

According to the Theorem 1, in order to show the mBR′ security of a key exchange protocol Π that is based on an IND-CCA public-key encryption scheme and computes a session key by hashing, we have to do the following.

- Show that in the case of a benign adversary the protocol completes correctly with a random key (see Section 2.2).
- Show that Π has strong partnering. We can always have this property if we add partnering information into the *session string* (see Section 2.2).
- Show that the cNR-mBR′ security of the related "*no-hashing*" protocol π is probabilistic polynomial time reducible to the hardness of the computational problem. First we check if both sides contribute nonces (very easily checked) and then we use the automatic tool by Cortier et al [12] to check occultness.

We do not have to show that given the decisional problem oracle of f, we can solve the session string decisional problem, because it is done by Lemma 5. Although there a number of steps, they can be done quickly or automatically. Checking the *occultness* property is the only difficult step but it can be automated.

Example 1. Suppose we want to prove the following key exchange protocol Π, which is based on Needham-Schroeder-Lowe protocol. There are parties A and B communicating as follows.

1. A $\rightarrow$ B: $\mathsf{Enc}_{P_B}(\mathsf{concat}(N_A, A))$
2. B $\rightarrow$ A: $\mathsf{Enc}_{P_A}(\mathsf{concat}(N_A, N_B, B))$

3. A $\rightarrow$ B: $\mathsf{Enc}_{P_B}(\mathsf{concat}(N_B))$

After that, A and B compute $sid = \{N_A, A\}_{P_B}, \{N_A, N_B, B\}_{P_A}, \{N_B\}_{P_B}$ and the session key $seskey_{\Pi} = hash(N_A, N_B, sid, A, B)$, where $hash(\cdot)$ is a hash function.

Π has mBR$'$ security because of the following.

- It is trivial to see that the protocol is functional. And because the session key is computed by hashing the concatenation of some uniformly chosen nonces, the session key is distributed uniformly.
- Π has strong partnering because we add (sid, A, B) into the session string according to the technique mentioned in Section 2.2.
- In the related protocol π, both parties contribute nonces. Furthermore, there is a mechanised proof of occultness for π by Cortier et al. [12].

5 Conclusion and Future Work

We have shown an approach of using an automatic technique designed originally for Dolev–Yao models to verify security of public-key-based key exchange protocols in a computational model. The full computational model is reduced to a simpler one first, before we apply a mechanisable technique to establish a security proof. Although the technique was first designed for checking a property in the Dolev–Yao model, we have shown that property also implies security in our simpler computational model. Therefore, the automatic technique proposed by Cortier et al. [12] can be used here to achieve a computationally sound security proof.

This work can be extended in some directions. Firstly, we want to know how the computational model can be extended, for example to modelling symmetric key encryption, while our approach remains applicable. Secondly, it may be possible to design an automatic technique to establish a security proof directly in the cNR-mBR$'$ model, e.g. using Hoare logic. This may allow us to treat more types of protocols than using the indirect method in this paper. Thirdly, because it is always hard to apply automatic techniques on full computational models, it would be interesting to find more modular approaches, e.g. reducing the complexity of models, and then designing automated proofs in the simpler models.

References

1. M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. *Lecture Notes in Computer Science*, pages 259–274, 2000.
2. M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated key exchange secure against dictionary attacks. Lecture Notes in Computer Science, pages 139–155. Springer, 2000.

3. M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Advances in Cryptology – CRYPTO93*, pages 232–249. Springer, 1993.
4. B. Blanchet. Automatic proof of strong secrecy for security protocols. In *2004 IEEE Symposium on Security and Privacy, 2004. Proceedings*, pages 86–100, 2004.
5. B. Blanchet. A computationally sound mechanized prover for security protocols. *IEEE Transactions on Dependable and Secure Computing*, 5(4):193–207, 2008.
6. R. Canetti and J. Herzog. Universally composable symbolic analysis of cryptographic protocols. In *Theory of Cryptography Conference (TCC06)*. Citeseer, 2006.
7. Ran Canetti and Sebastian Gajek. Universally composable symbolic analysis of Diffie–Hellman based key exchange. Cryptology ePrint Archive, Report 2010/303, 2010. `http://eprint.iacr.org/`.
8. Ran Canetti and Hugo Krawczyk. Universally composable notions of key exchange and secure channels. In Lars R. Knudsen, editor, *Advances in Cryptology - EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 337–351. Springer, 2002.
9. Y. Chevalier and L. Vigneron. A tool for lazy verification of security protocols. In *Proceedings of ASE*, volume 1, pages 373–376. Citeseer, 2001.
10. K.K. Choo, C. Boyd, and Y. Hitchcock. Errors in computational complexity proofs for protocols. *Advances in Cryptology-ASIACRYPT 2005*, pages 624–643, 2005.
11. K.K.R. Choo, C. Boyd, Y. Hitchcock, and G. Maitland. On session identifiers in provably secure protocols. *Security in Communication Networks*, pages 351–366, 2004.
12. V. Cortier, J. Millen, and H. Rueß. Proving secrecy is easy enough. In *Proceedings of the 14th IEEE workshop on Computer Security Foundations*, page 97. IEEE Computer Society, 2001.
13. V. Cortier and B. Warinschi. Computationally sound, automated proofs for security protocols. *Programming Languages and Systems*, pages 157–171.
14. J. Courant, M. Daubignard, C. Ene, P. Lafourcade, and Y. Lakhnech. Towards automated proofs for asymmetric encryption schemes in the random oracle model. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, pages 371–380. ACM, 2008.
15. A. Datta, A. Derek, JC Mitchell, and B. Warinschi. Computationally sound compositional logic for key exchange protocols. In *19th IEEE Computer Security Foundations Workshop, 2006*, page 14, 2006.
16. D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.
17. M. Gagné, P. Lafourcade, Y. Lakhnech, and R. Safavi-Naini. Automated security proof for symmetric encryption modes. *Advances in Computer Science-ASIAN 2009. Information Security and Privacy*, pages 39–53.
18. C. Kudla and K. Paterson. Modular security proofs for key agreement protocols. *Advances in Cryptology-ASIACRYPT 2005*, pages 549–565.
19. G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. *Tools and Algorithms for the Construction and Analysis of Systems*, pages 147–166, 1996.
20. D. Micciancio and B. Warinschi. Soundness of formal encryption in the presence of active adversaries. *Theory of Cryptography*, pages 133–151, 2004.
21. J. Millen and H. Rueß. Protocol-independent secrecy. In *2000 IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2000.
22. T. Okamoto and D. Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. *Lecture Notes in Computer Science*, 1992:104–118, 2001.